A Web Services Component Discovery and Deployment Architecture for Simulation Model Reuse

David Bell
Navonil Mustafee
Sergio de Cesare
Mark Lycett
Simon J. E. Taylor
School of Information System, Computing and Mathematics
St. Johns,
Brunel University,
Uxbridge, UB8 3PH, UK.
+44 (0) 1895 203397
david.bell@brunel.ac.uk, navonil.mustafee@brunel.ac.uk, sergio.decesare@brunel.ac.uk,
mark.lycett@brunel.ac.uk, simon.taylor@brunel.ac.uk

Keywords:
Simulation components, Ontology, Model Integration.

ABSTRACT: *CSPs are widely used in industry, although have yet to operate across organizational boundaries. Reuse across organizations is restricted by the same semantic issues that restrict the inter-organization use of web services. The current representations of web components are predominantly syntactic in nature lacking the fundamental semantic underpinning required to support discovery on the emerging semantic web. Semantic models, in the form of ontology, utilized by web service discovery and deployment architecture provide one approach to support simulation model reuse. Semantic interoperation is achieved through the use of simulation component ontology to identify required components at varying levels of granularity (including both abstract and specialized components). Selected simulation components are loaded into a CSP, modified according to the requirements of the new model and executed. The paper presents the development carried out within CSPI-PDG and Fluidity Group at Brunel University, of an ontology, connector software and web service discovery architecture. The ontology is extracted from simulation scenarios involving airport, restaurant and kitchen service suppliers. The ontology engineering framework and discovery architecture provide a novel approach to inter-organization simulation, adopting a less intrusive interface between participants. Although specific to CSPs the work has wider implications for the simulation community.*

1. Introduction

Commercial-off-the-shelf (COTS) simulation packages (CSPs) offer an interactive and visual model development environment for creating computer models of existing and proposed systems and experimenting with the same. Simulation practitioners in industry extensively use CSPs like Simul8, Witness, AnyLogic, AutoMod and Arena to model their simulations. These packages allow reuse of standard simulation components like workstations, queues, conveyors, resources etc. and thereby provide the building blocks which facilitate creation of larger models. As these models grow larger and more complex the prospect of simulation model reuse is appealing as it has the potential to reduce the time and cost incurred in developing future models. An extension of model reusability is the concept of separate development and user groups, whereby models are developed and validated by one group and then used to specify simulations by another group [23]. In this paper we look at the *discovery and import* of CSP-created models across organizational boundaries in the context of supply chains, thus enabling the development and user groups to exist in different organizations. This approach does not allow model information hiding between enterprises and contrasts with the *distributed simulation* approach to model reuse which allows an organization to hide model

specific information and data from the other participants. A short discussion on supply chains and the distributed simulation approach follows.

Supply Chain Management (SCM) consists of a series of tasks like manufacturing, transport and distribution that are undertaken by organizations with an aim of delivering products to their customers. Simulation of the supply chain can identify manufacturing bottlenecks, resources required for on time delivery, adequate stock levels for distribution etc. and help improve the performance of the underlying supply chain. Each organization that forms a part of the supply chain normally develops models that simulate their own part of the supply chain using CSPs [26]. Assuming that all necessary individual simulation components are now available the question is how do we link them together? Distributed simulation offers one such solution. Distributed simulation can be defined as the distribution of the execution of a single run of a simulation program across multiple processors [31]. It allows each organization to run its model in its own site (thereby encapsulating model details within the organization itself) and participating with other sites through information exchange using distributed simulation middleware [27]. [21, 29, 22, 30] are examples of successful distributed simulation using CSPs. There is a growing body of research dedicated to creating distributed simulation with CSPs and the High Level Architecture (HLA), the IEEE 1516 standard for distributed simulation. In an attempt to unify this research COTS Simulation Package Interoperability Product Development Group (CSPI-PDG), a Simulation Interoperability Standards Organization (SISO) standardization group that began operation in October 2004 (http://www.cspi-pdg.org/).

The distributed simulation approach to model reusability in the context of CSPs faces the following challenges. Firstly, a lack of widespread demand for distributed simulation in industry has meant that the CSP vendors have not currently incorporated distributed simulation support into their products. Consequently, the organizations that want to use this approach do not have readymade solutions. Secondly, research projects that aim to create CSP based distributed simulation do not have access to its source code are thus limited by the functionality offered by the vendor. Thirdly, execution of a distributed simulation tends to be much slower than traditional standalone simulation. For example, the straightforward use of the conservative HLA time advance mechanisms results in a simulation that runs extremely slowly, at times a few factors slower that its corresponding sequential runs [22]. In order to progress, these issues have to be resolved before industry can fully benefit from the application of CSP based distributed simulation. In the meantime it is worth investigating other approaches enabling supply chain simulation across organizational boundaries.

Our discovery and import approach to model reuse in the context of CSPs offer an alternative to the distributed simulation approach. By discovery we mean that individual simulation models, which are created by organizations to model their activity in the supply chain, are discovered from among an inter-organizational repository of models spread across the organizations. The selected models are then loaded into a CSP, modified according to the requirements of the new model and executed. We believe that our approach to enabling CSP based supply chain simulation has a lighter touch with much fewer technical barriers. It also requires minimal CSP vendor intervention when compared to the distributed approach.

Our vision is a web of SC models that are accessible to the practitioner. The current representations of web components are predominantly syntactic in nature lacking the fundamental semantic underpinning required to support discovery on the emerging semantic web [1] Semantic models, in the form of ontology, utilized by web service discovery and deployment architectures provide one approach to support simulation model reuse. Improved component reuse supported by ontological models has already been proposed in simulation [2]. When considering COTS Simulation packages, intrusive activities are not possible when dealing with packaged software as only import or export capabilities are achievable. The tools of the semantic web provide a means to construct external description of the CSP models. This external description, or ontology, can then be used to support the reuse of simulation components (SCs). Consider a scenario where a large multinational organisation uses CSPs to model many of its business activities. Two human process are undertaken when a simulation is required – the creation of the model and its execution. In order to fully utilise the capabilities within the organisation we propose that *model parts* can be reused more effectively, better utilising the expertise within distinct models. In order to support the reuse, methods for describing the models then enable semantic discovery are proposed. The system supports the discovery of specific model components and their loading into the COTS simulation package. Semantic interoperation is achieved through the use of a simulation component ontology to identify required components at varying levels of granularity (including both abstract and specialized components). Once

selected, simulation components are loaded into a CSP, modified according to the requirements of the new model and executed. The ontology is derived from existing CSP Simulation Components (SCs) and is contrasted to current simulation ontology.

The paper proposes that the evolutionary construction of domain grounded SC ontology better supports the semantic discovery of SCs. In addition, when combined with hard simulation semantics (such as state etc.), concepts from both vocabularies provide improved matching terms.

The paper is organized as follows. Section 2 presents a summary of pertinent literature including a summary of semantic web and ontologies. Section 3 describes the DESC ontology and the process undertaken to engineer it. Section 4 covers the software tools that use the DESC ontology – the semantic search and component integration software. A conclusion summarizes the work presented.

2. Related Literature

Two communities of research are relevant to the work presented here: (1) Semantic web services and (2) the grid resource discovery. Both provide an insight into the decoupling of component models from their execution environment and are used for discovery and synthesis. Semantic search has been applied to both topics with a common reliance on knowledge – referred to as service ontology. Ontology itself is a specification of a representational vocabulary for a shared domain of discourse – with definitions of classes, relations, functions, and other objects [3]. It is an explicit specification of a conceptualization. The term is borrowed from philosophy, where an Ontology is a systematic account of existence [3]. In borrowing the term ontology and placing it into an engineering discipline, two distinct usage types emerge in the creation of these specifications: The theoretic (deductive) approach and the pragmatic (inductive approach) [35]. It is the pragmatic approach that is adopted in this paper – focusing on the engineering of knowledge from CSP models.

The semantic web provides the knowledge structure and reasoning about a web of models and the grid because our vision is a grid of CSPs that are able to execute discovered models. The semantic web [4] aims to uncover knowledge about domains so as to better support discovery, integration and understanding of resident objects. Semantic web services SWS refine this vision [5] making web services "computer-interpretable, use apparent, and agent-ready". With this web of services comes a need to describe explicitly and in a form able to be read by computer.

Current intersections between web services and the semantic web have delivered a diverse body of research. The agent community [5-7] has recognized the benefit of ontology if computer-to-computer web architectures are to be achieved. Combining service and domain ontology is seen as a key to achieving service synthesis [8]. Work on service ontology is currently centered on OWL-S and WSMO groups. Recognizing the progress, by the DAML Consortium and others, attention has moved from the ontology languages to specific application to services. A discussion of semantic web services would not be complete without coverage of the OWL-S upper ontology model (WSMO is less mature at this time although similar in nature). The OWL-S high level model describes the relationship between the differing service decompositions (see Figure 1) [8, 9]. A resource provides a service that is represented by the ServiceProfile, described by the ServiceModel and supported by the ServiceGrounding. Generally, the profile describes the service in a high level way (enough to discover the service), the model describes the detail of how it works and can be used to: (1) perform more in-depth analysis of whether the service meets a need, (2) to compose service descriptions from multiple services to perform a specific task, (3) during enactment, to co-ordinate activities from participants and (4) to monitor execution [9]. The service grounding details practical access and has converged with WSDL.

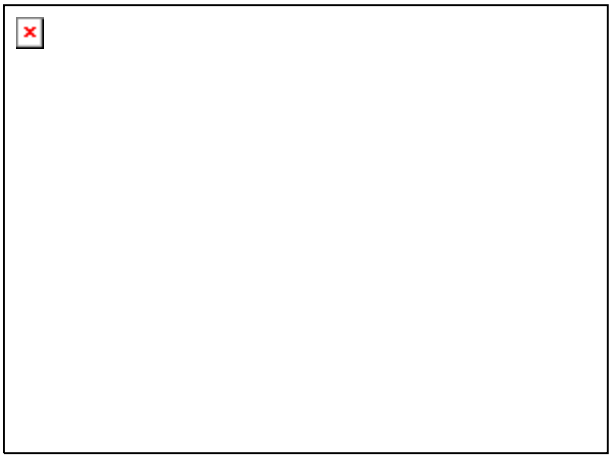

Figure 1: OWL-S Upper Ontology

OWL-S (and WSMO) [10]provide generalized models for describing services. Others have identified the need for specialized common concepts within a web service context [10-14], with one example being quality of service. These concepts represent glue homogenizing a wealth of asymmetrically described

web resources. New issues become pertinent in a semantic web of "great number of small ontological components consisting largely of pointers to each other" [15]. This semantic web service environment, with recognition of the need to combine service and domain ontology, warrants research that identifies practical approaches for businesses to combine the service ontology with existing or new domain ontology. The foremost question in semantic service orientation is how best this should be undertaken in the context of simulation.

Transporting this vision to a simulation environment with a web of simulation components has several challenges. Combining distributed SCs models into a new model requires that they are discovered. Consequently, explicit, computer readable knowledge is required for such search tasks. Knowledge in the form of ontologies has already been applied to simulation [16] with work by the University of Florida on simulation translation and University of Georgia on a taxonomy of simulation objects called DeMO. DeMO provides a precise description of simulation models with hard semantics. In order to realize a vision for SCs similar to that of SWS requires that the domain being simulated is represented explicitly (an OWL ontology [17]). The DeMO ontology [16] is an upper ontology that details events, activities and processes. Hard semantics work perfectly if all stakeholders adopt the single model. If this is not the case, and with only the CSP SCs, a transformation directly to such a model will likely miss tacit domain concepts that may help any subsequent SC search activity.

The eXtensible Modeling and Simulation Framework (XMSF) is defined as a set of composable standards, profiles and recommended practices for web-based modeling and simulation. XMSF prescribes the use of ontologies for the definition, approval and interoperability of complimentary taxonomies that may be applied across multiple simulation domains [20]. In military modeling and simulation, the study of ontology is recognized as important in developing techniques that would allow semantic interoperability between simulation systems and to this effect ontology of C2IEDM (Command and Control Information Exchange Data Model) has been created to further studies on enabling interchange of data between two or more systems [34]. Work is also underway for creating an ontology for physics which would represent physics-based model semantics in modeling and simulation. Its intension is to capture the concepts of physical theories in a formal language so as to support various forms of automated processing that are currently not supported [24]. An ontology for the representation of data pertaining to Synthetic Environment called sedOnto (Synthetic Environment Data Representation Ontology) has been proposed [20]. Finally, ongoing work is looking into establishing an ontology for BML, an unambiguous language to command and control forces and equipment [33].

3. Simulation Component Ontology

3.1 Requirement for Semantic Search

The globalization of many organisations and industries often result in a fragmentation and heterogeneity of knowledge produced by its domain experts. In order to synthesize the most appropriate knowledge in a model, the best available model parts must first be found. Syntactic or taxonomic approaches limit the precision in which SCs can be related to the domain. Typical issues are that a component may not fit neatly into a prescribed category or simple use of synonyms to describe the component.

3.2 DESC Ontology

The Discrete Event Simulation Component (DESC) ontology resulted from two distinct research activities: (1) The transformation of CSP models into OWL ontology files and (2) semantic search scenarios being carried out against the OWL files. Snapshots of DeMO and DESC ontologies are presented in figures 2 and 3. The differences are apparent with DeMO focusing on the component properties and DESC on the component in relation to the domain. Links between the two models are achieved through referencing the DeMO:ModelComponent from the DESC:SimulationConcept when it relates to an available component model. Additionally, the DeMO ontology is imported into Protégé in order to use it's classes as properties of the DESC ontology (for example, when describing a business concept that is a specific *state* or *activity* in the simulation).

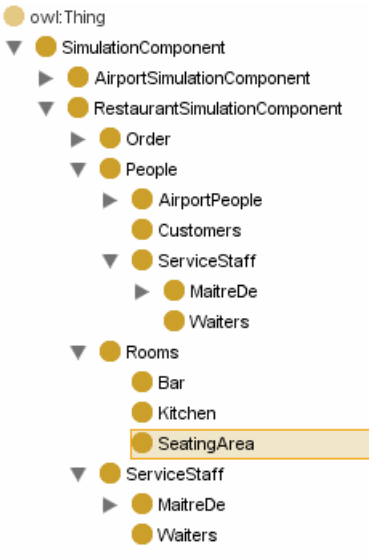

Figure 2. DESC-Restaurant Ontology Structure

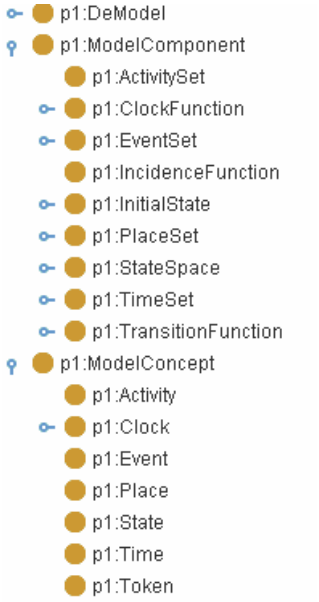

Figure 3. DeMO Ontology Structure

The ontology was created using the Protégé tool from Stamford University (with Owl plugins) (http://protege.stanford.edu/). A decision was made to ground the ontology in existing SCs as opposed to using particular service ontology such as OWL-S or WSMO.

3.3 Ontology Engineering

A number of activities were carried out to transform three CSP models into ontological form – OWL files. The process included the decoupling of the SCs from the model by placing disctinct component models into a web based component library (URI accessible). The activities carried out, in framework form, are detailed in Table 1. The framework evolved as each CSP model was deconstructed and transformed into ontology classes (including relations to dependent or related classes). Realization of the need for a DESC ontology resulted from this process – which included the adoption of DeMO for hard component semantics.

Activities	Description	Impact
Component Extraction	Specific components are extracted to form distinct models. These are stored in the DESC library (a standard web server).	▪ CSP models ▪ SC Models
Component Typing	A new class is added to the OWL ontology to represent the SC. Similar classes are grouped under a type.	▪ OWL Classes
Component Dependency Models	Extended DeMO properties are used to define dependencies between services. E.g. StateDependency. Reference DeMO concepts when describing business properties (e.g. ThinkingTable has a DeMO state property). New classes and properties are created for previously implied activities etc. (e.g. Serving is a created from an analysis of table in ordering and eating).	▪ OWL Properties ▪ New OWL Classes and properties implied from the model
Ontology Testing	The finalized ontology is loaded into the SEDI4G server and several search tasks are undertaken.	▪ DESC OWL File

Table 1: Process for deriving semantic content from CSP Models

The ontology engineering process resulted in DESC-RESTAURANT (Figure 2), DESC-KITCHEN and DESC-AIRPORT models (OWL Files). Each provided more component returns as concept inferencing was able to traverse the concept tree and return additional suitable candidates. The process undertaken to engineer the domain simulation ontology provides the basis for subsequent modelers to reference

and extend the domain ontology; thus achieving richer search results and evolving large component ontology. The ontology engineering process systematically analyses the CSP model, of which figure 4 is an example.

Figure 4. Simul8 Model

4. Discovery and Import of Simulation Components

Our *discovery and import* approach aimed at CSP model reuse enables us to (1) semantically search for the desired simulation models and (2) parse and import the identified models into a simulation package. For our demo application we have used CSP Simul8. Simul8 enables users to rapidly construct accurate, flexible and robust simulations using an easy-to-use visual modeling interface [13]. However, our discovery and import architecture has the potential to support any CSP that allows an external program to perform basic operations such as opening the CSP and loading a model through its Component Object Model (COM) interface. COM is a Microsoft technology that allows different software components to communicate with each other by means of interfaces [14]. The discovery component of our architecture (described in section 4.1) can be used with very little change to support other CSPs. The parse and import component, however, would require implementation of a CSP specific parser (described in section 4.2) and cannot be reused.

4.1 Design of Component Discovery System

The component discovery system is an extension of the SEDI4G architecture [18]. Extending the application to support SC descriptions as well as grid services required only minor configuration changes to support the new OWL DESC ontology. The semantic discovery system shown is figure 5 comprises a set of web services (SCVD, SDCS and SMAS).

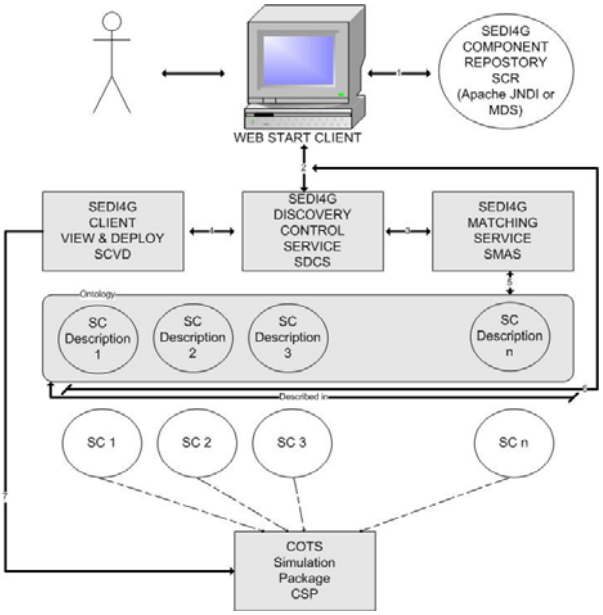

Figure 5.Discovery Architecture

The discovery process begins by identifying the web services and ontology required to carry out semantic search. The choices are directed by the ontology size and service placement on the network (represented by the grey flexible services and data in Fig. 1). Thus, Step 1 involves the selection of which discovery control service (SDCS), knowledge base and matching service best fit the user requirement – specified as text strings. This information is sent to SDCS together with the search parameters (2). SDCS then calls the KB based matching service SMAS (based on OWLJessKB (http://edge.cs.drexel.edu/assemblies/software/owljesskb/)) (3) that in turn loads the KB and rules (5). The maching is carried out and returned to SDCS for use in one of the client components (4). The SDCS service can optionally provide the resource properties, the dynamic state of each service, alongside the service choices (6). Finally the returned components are displayed in a web start client (SCSV holding the component options on the server side) allowing selected components to be deployed into the CSP. The deployment is simple in nature, loading server side XML into the CSP. A more robust solution would provide transformation capabilities as has been done at Florida [16].

The matching algorithm is semantic and uses an ontology and a reasoning engine. The assumption in this paper is that an ontology is a catalogue of the types of “things”; derived from existing simulation models. Types in the ontology represent the predicates, word meanings, or concept and relation types of the

language when used to discuss topics in the domain [18] – in this paper these are SCs.

To summarize, the matching algorithm comprises two steps; the initialization of the knowledge base and the search. During the initialization phase the ontology is loaded transforming ontological classes into facts that have rules applied using the Rete algorithm [19]. During the search inferences are made from the facts (using Jess queries) identifying semantically matched SCs. For example, when searching for a component to simulate a restaurant table – several are returned that model different states.

4.2 Design of CSP Model Parser and Importer

The discovery architecture detailed in the previous section is used by the CSP Model Parser and Importer (CMPI) software to conduct a semantic search for existing models. This search is conducted by calling a web service defined in the component discovery architecture, which takes a search string as parameter and returns an enumeration of uniquely identified name (URN) and corresponding unique resource locator (URL) for each model returned by the matching algorithm. CMPI then provides the user an option to (1) download the models into the local system for introspection or (2) import it directly into the new model being built through reuse of the discovered components.

In case the user chooses option (1) the model can be downloaded into the local system by clicking on the URL, as with any file download from the Internet. The file downloaded is an XML representation of the Simul8 model which was discovered.

If the user chooses option (2) the URN is passed as a parameter to yet another web service which returns the XML representation of the model as a SOAP attachment. The nature of this web service is synchronous and this allows the CMPI to block further execution of the code until the XML file has been received.

The merging of the existing model (being built through reuse of discovered models and model components) with the new model requires a CSP specific parsing operation. Since both the models in question have an XML representation we employ a crude text parsing mechanism which traverses through the XML hierarchy of these models and outputs a third XML file containing assimilated results from both. This new XML file is now loaded into the CSP and the user is presented with the overall model. It should be added that the text parsing mechanism is heavily dependent on the Simul8 specific knowledge and has not yet be fully perfected. However, this is not a major problem because a model can be opened in Simul8, copied into the clipboard and pasted into another Simul8 model. This solution would alleviate the need for a model parser.

The CMPI software is written in Java and it uses the Simul8 COM interface to interact with Simul8 using Java Native Technology [32]. CMPI invokes web service calls to communicate with the component discovery system. It also includes a CSP specific parser component which, as has been discussed in the previous paragraph, can be considered optional. The architecture and dependencies of CMPI is shown in Figure 6.

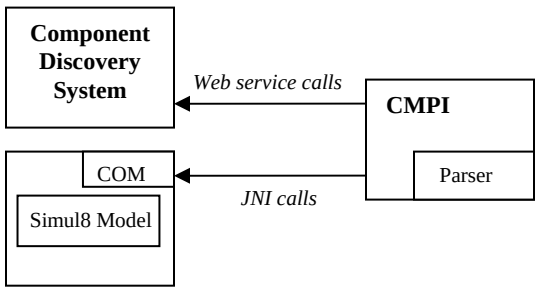

Figure 6. Architecture of dependencies of CMPI

5. Conclusion

The paper presents a novel approach to CSP model reuse using a simulation component ontology and semantic search architecture. The approach to modeling simulation components focuses on the domain in which they are modeling. In relating each component to a type collection and each other enables the search process to better identify likely semantic matches. Several Simul8 models are transformed into OWL ontologies and then used by a web service based semantic search and component deployment architecture. The research has demonstrated: (1) a new, lighter approach to CSP model reuse and (2) the benefits of semantic search to this field of research.

6. References

[1] D. Bell, S. de Cesare and M. Lycett, "Semantic transformation of web services," in *OnTheMove*

2005 (SWWS 2005 Workshop), 2005, pp. 856-865.

[2] J. A. Miller, P. A. Fishwick, G. Baramidze and A. P. Sheth, "Ontologies for Modeling and Simulation: An Extensible Framework (Under Revision)," *TOMACS*, 2006.

[3] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, pp. 199-220, 1993.

[4] T. Berners-Lee, J. Hendler and O. Lassila, "The Semantic Web," *Sci. Am.*, vol. 284, pp. 34-43, 2001. 2001.

[5] S. A. McIlraith, T. C. Son and H. L. Zeng, "Semantic Web services," *IEEE INTELLIGENT SYSTEMS & THEIR APPLICATIONS*, vol. 16, pp. p46-53,

[6] N. Gibbins, S. Harris and N. Shadbolt, "Agent-based semantic web services," in *Proceedings of the 12th International Conference on World Wide Web* Anonymous Budapest, Hungary: ACM Press, 2003, pp. 710-717.

[7] D. Martin, A. J. Cheyer and D. B. Moran, "The Open Agent Architecture: A Framework for Building distribted Software Systems," *Appl. Artif. Intell.*, vol. 13, pp. 91-128, January-March 1999. 1999.

[8] L. Chen, N. R. Shadbolt, C. Goble, F. Tao, S. J. Cox, C. Puleston and P. Smart, "Towards a knowledge-based approach to semantic service composition," in *Second International Semantic Web Conference (ISWC2003)*, 2003,

[9] A. Ankolekar, M. Burstein, J. R. Hobbs, O. Lassila, D. Martin, D. McDermott, S. A. McIlraith, S. Narayanan, M. Paolucci, T. Payne and K. Sycara, "DAML-S: Semantic markup ForWeb services," in *International Semantic Web Working Symposium (SWWS)*, 2001, pp. 348-363.

[10] R. Lara, D. Roman, A. Polleres and D. Fensel, "A conceptual comparison of WSMO and OWL-S," in *Web Services: European Conference, ECOWS 2004*, 2004, pp. 254-269.

[11] J. Cardoso and A. Sheth, "Semantic e-workflow composition," *J. Intell. Inf. Syst.*, vol. 21, pp. 191-225, Nov. 2003.

[12] M. Paolucci, T. Kawamura, T. R. Payne and K. Sycara, "Semantic matching of web services capabilities," in *Semantic Web - Iswc 2002* , vol. 2342; 2342, Anonymous Berlin: SPRINGER-VERLAG BERLIN, 2002, pp. 333-347.

[13] F. Curbera, M. Duftler, R. Khalaf, W. Nagy, N. Mukhi and S. Weerawarana, "Unraveling the Web services Web - An introduction to SOAP, WSDL, and UDDI," *IEEE Internet Comput.*, vol. 6, pp. 86-93, Mar-Apr. 2002.

[14] V. Tosic, B. Esfandiari, B. Pagurek and K. Patel, "On requirements for ontologies in management of web services," in *Web Services, E-Business, and the Semantic Web* , vol. 2512; 2512, Anonymous Berlin: SPRINGER-VERLAG BERLIN, 2002, pp. 237-247.

[15] J. Hendler, "Agents and the Semantic Web," *Intelligent Systems, IEEE [See also IEEE Intelligent Systems and their Applications]*, vol. 16, pp. 30-37, 2001.

[16] P. A. Fishwick and J. A. Miller, "Ontologies for modeling and simulation: Issues and approaches," in 2004, pp. 259--264.

[17] W3C, "Web Ontology Language," 2005.

[18] D. Bell and S. A. Ludwig, "Grid Service Discovery in the Financial Markets Sector," *JCIT*, vol. 13, pp. 265-170, 2005.

[19] C. L. Forgy, "Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problems," *Artif. Intell.*, vol. 19, pp. 17-37, 1982.

[20] M. Bhatt, W. Rahayu and G. Sterling, "sedOnto: A web enabled ontology for synthetic environment representation based on the SEDRIS specification," in *Fall Simulation Interoperability Workshop*,

[21] C. A. Boer, A. Verbraeck and H. P. M. Veeke, "Distributed simulation of complex systems: Application in container handling," in *European Simulation Interoperability Workshop*, 2002,

[22] Boon Ping Gan, M. Yoke, H. Low, Xiaoguang Wang and S. J. Turner, "Using manufacturing

process flow for time synchronization in HLA-based simulation," in *Ninth IEEE International Symposium on Distributed Simulation and Real-Time Applications*, 2005, pp. 148-157.

[23] B. J. Bortscheller and E. T. Saulnier, "Model reusability in a graphical simulation package," in *WSC '92: Proceedings of the 24th Conference on Winter Simulation*, 1992, pp. 764-772.

[24] J. B. Collins, "Standardizing an ontology of physics for modeling and simulation," in *Fall Simulation Interoperability Workshop*,

[25] K. H. Concannon, K. I. Hunter and J. M. Tremble, "Dynamic scheduling II: SIMUL8-planner simulation-based planning and scheduling," in *WSC '03: Proceedings of the 35th Conference on Winter Simulation*, 2003, pp. 1488-1493.

[26] R. M. Fujimoto, *Parallel and Distributed Simulation Systems*. John Wiley & Sons, 2000,

[27] B. P. Gan, L. Liu, S. Jain, S. J. Turner, W. Cai and W. Hsu, "Manufacturing supply chain management: Distributed supply chain simulation across enterprise boundaries," in *WSC '00: Proceedings of the 32nd Conference on Winter Simulation*, 2000, pp. 1245-1251.

[28] D. N. Gray, J. Hotchkiss, S. LaForge, A. Shalit and T. Weinberg, "Modern languages and Microsoft's component object model," *Commun ACM*, vol. 41, pp. 55-65, 1998.

[29] K. Mertins, M. Rabe and F. Jaekel, "Neutral template libraries for efficient distributed simulation within a manufacturing system engineering platform," in *WSC '00: Proceedings of the 32nd Conference on Winter Simulation*, 2000, pp. 1549-1557.

[30] N. Mustafee and S. J. E. Taylor, "Investigating distributed simulation with COTS simulation packages: Experiences with Simul8 and the HLA," in *2006 Operational Research Society Simulation Workshop (SW06)*, 2006, pp. 33-42.

[31] S. J. E. Taylor, R. Sudra, T. Janahan, G. Tan and J. Ladbrook, "Towards COTS distributed simulation using GRIDS," in *WSC '01: Proceedings of the 33nd Conference on Winter Simulation*, 2001, pp. 1372-1379.

[32] http://java.sun.com/j2se/1.4.2/docs/guide/jni/, Sun Microsystems Limited.(2003).Java Native Interface., vol. 2006,

[33] A. Tolk and C. Blais, "Taxonomies, ontologies, and battle management languages – recommendations for the coalition BML study group, spring simulation interoperability workshop," in

[34] A. Tolk and C. Turnitsa, "Ontology of the C2IEDM - further studies to enable semantic interoperability," in *Fall Simulation Interoperability Workshop*.

[35] G. Geerts and W. E. McCarthy, "An accounting object infrastructure for knowledge-based enterprise models," *IEEE Intelligent Systems & their Applications*, vol. 7, 1999.